REMARKS

Initially, Applicant expresses appreciation to the Examiner for the courtesies extended in the recent telephonic discussion of this case with Applicant's representative. The amendments and remarks presented herein are consistent with those discussions. Accordingly, entry of this amendment and reconsideration of the pending claims is respectfully requested.

The Office Action, mailed April 6, 2007 considered and rejected claims 1-5, 7-18, 21-28 and 30-38. Claims 1-5, 7-18, 21-28 and 30-38 were rejected under 35 U.S.C. 103(a) as being unpatentable over *Gombocz* (U.S. Publ. No. 2002/0156792) in view of *Wong* (U.S. Patent No. 6,889,229).[1]

By this paper, claims 1, 10, 17, 21, 25 and 27 have been amended, while no claims have been added or cancelled. Accordingly, following this paper, claims 1-5, 7-18, 21-28 and 30-38 remain pending, of which claims 1, 10, 17, 21, 25 and 37 are the only independent claims at issue.

As discussed previously with the Examiner, Applicant's claims generally relate to extending the use of data-types between different middle-tier servers in a multi-tier server system, and without transferring the data-type information between middle-tier servers in a peer-to-peer system. As recited in claim 1, for example, a method is disclosed for deploying one or more data types from a back end server of the multi-tier server system to any of a plurality of middle tier servers so as to maintain consistency and compatibility in the definitions of the data types and in the code associated with each data type stored on each of the middle tier servers. As recited, the method can include an act of creating a special table in a database on the back end server, the special table including one or more fields for storing data identifying data types used by the plurality of middle tier servers as well as code for enabling the use of each of the data types. The database of the back end server acts as a repository for each data type used by any of the middle tier servers, such that the back end server acts as a single and centralized source from which each of the middle tier servers obtains all data types used by any other of the plurality of middle tier servers as well as the corresponding code for use of the data types. Accordingly, the single and centralized source from which each middle-tier server obtains all data types is at a

[1] Although the prior art status of the cited art is not being challenged at this time, Applicant reserves the right to challenge the prior art status of the cited art at any appropriate time, should it arise. Accordingly, any arguments and amendments made herein should not be construed as acquiescing to any prior art status of the cited art.

different tier than the middle tier-servers to which the data type is being extended. An extended assembly that corresponds to the data type to be deployed is also obtained which includes data from the special table such as data identifying the data type, one or more definitions, and code for processing the data. The extended assembly is also transmitted to at least one of the middle tier servers.[2]

As discussed with the Examiner, while the cited references generally relate to transferring data between computer devices, Applicant respectfully submits that they fail to disclose or suggest, whether cited individually or in combination, each and every limitation of the pending claims. (*See* M.P.E.P. §§ 2142, 2143). For example, among other things, the cited references fail to disclose or suggest wherein in a multi-tier server having a back-end server serving a plurality of middle tier servers, the back end server acts as a single and centralized source from which all middle tier servers can obtain information relating to data types, and such that each of the multiple middle tier servers obtains all data types from the single, centralized source which operates at a different tier than the plurality of middle tier servers.

For example, *Gombocz* discloses an object handling system for intelligently handling data according to dynamic application needs. For example, an object translation engine (OTE) is described which connects to multiple external databases and to other optional components. (¶ 193). The OTE is further connected with an application/database generator (ADG) and master query component (MQC) for translating requested data from one type to a different type. (¶¶ 193, 194).

Specifically, the OTE component includes methods for bi-directional interchange with components and access interfaces, and provides methods for processing heterogeneous and incompatible data into recognized and understandable data. (¶ 194). Thus, an application which uses data of a first type can request data and, even if the requested data is of an incompatible data type, the OTE can translate the data to a recognized type. (¶¶ 194, 195).

Accordingly, *Gombocz* discloses that an application can use an OTE to obtain data, even though the data is stored in a different format or is of a different type. *Gombocz* fails to disclose, however, that the requesting computer even receives the type information. Instead, the data is merely obtained in a translated format which is compatible with type information already known

[2] Independent claims 10, 17, 21, 25 and 37 generally correspond to the method of claim 1. Accordingly, the discussion herein with regard to claim 1 applies equally to each of claims 10, 17, 21, 25 and 37.

to the application. In short, *Gombocz* discloses that data is requested and converted between data types, but fails to disclose that data types are sent from a central repository to multiple middle-tier servers.

Moreover, although the data is requested and transferred in *Gombocz*, the reference fails to disclose that all data type information is obtained from middle tier servers from a back end server which operates as single and centralized source, and at a different tier than the requesting middle tier servers. Indeed, as discussed with respect to Figure 4, *Gombocz* discloses that a variety of computers are connected with peer-to-peer and an any-to-any connectivity, such that the intelligent object handler system can obtain requested information from any connected source, rather than from any single source. (¶¶ 67, 68). Indeed, the Office appears to have acknowledged this deficiency in *Gombocz* inasmuch as it notes that *Gombocz* fails to disclose a "back end server acting as a single and centralized source from which all middle tier servers obtain all data types used by any other of the plurality of middle tier servers."

Recognizing this deficiency, the Office Action then relies upon the *Wong* reference. For at least the reasons discussed with the Examiner, Applicant respectfully believes that reliance on *Wong* is in error. In particular, while *Wong* discloses transferring data type information between computers, *Wong* expressly discloses that the system operates using "peer-to-peer replication of objects in a relational database." Accordingly, *Wong* expressly teaches that objects are replicated between computing systems at the same tier, rather than from a back end server operating at a different tier and as the single source of data types.

More particularly, *Wong* (see Fig. 1) describes a method for peer-to-peer replication of objects between various nodes connected over a network. Disclosed is a replication process for replicating user-defined objects to make them available to other nodes. (Col. 1, ll. 7-10). In *Wong*, users may define classes and generate, store and receive multiple user-defined objects based on each class at a respective computer used by the user. (Col. 1, ll. 39-43). If data is shared with other users on a network, a database may be copied from one node onto a new node that does not have a copy of the database. (Col. 2, ll. 53-56). According to *Wong*, the node on which a user-defined object is located creates a replication group of objects, including any user-defined objects. (Col. 6, ll. 41-56; Col. 8, ll. 25-43). Thereafter, a database server on the node copies data defining the user-defined object to a data structure on a second node. (Col. 9, ll. 25-31). Subsequently, a database server routine replicates the data by first copying the name of the

user-defined object from the first node to the new node. (Col. 10, ll. 31-35). Data defining the user-defined object is then copied to the new node, and data defining the database object is then copied from the data dictionary. (Col. 10, ll. 40-45, 52-60). A data dictionary is an object that includes, for example, the name of a particular table, the type of each column in the table, and the name of a user-defined object. (Col. 6, ll. 31-41; Col. 7, ll. 44-46). Finally, the new node may instantiate the database object based on data in the data dictionary, thereby obtaining a substantially identical database object from a peer node.

In contrast, and as recited in the claims herein, Applicant's claimed methods and computer program products are directed to deploying one or more data types from a back end server storing the data type information in a database, such that the back end server acts as a single and centralized source from which all middle tier servers obtain all data types used by other middle tier servers on the system and the corresponding code required to enable use of each of the data types, and such that each of the middle tier servers obtains all data types from a single and centralized source operating at a different tier than the middle tier servers.

Applicant's claimed method thus includes transferring data type data stored by a single, centralized source that operates at a different tier than nodes to which the data is extended. Accordingly, the disclosed claims exclude a peer-to-peer or any-to-any system in which data types can be moved between peers or between any computers. *Wong* specifically teaches the transfer of user-defined objects, such that each node can potentially have different objects, depending on the user of the node. As a result, each node disclosed in *Wong* has its own, unique user-defined objects and data structures which each node then replicates on other nodes which are parallel in the system hierarchy. Thus, multiple nodes operating at the same tier replicate their unique user-defined objects to other equally situated nodes. Applicant's claimed invention, in contrast, transmits all data types from a single and centralized back end server which operates at a different tier than the recipient systems.

In view of the foregoing and the other discussions with the Examiner, Applicant respectfully submits that the other rejections to the claims are now moot and do not, therefore, need to be addressed individually at this time. It will be appreciated, however, that this should not be construed as Applicant acquiescing to any of the purported teachings or assertions made in the last action regarding the cited art or the pending application, including any official notice.

Instead, Applicant reserves the right to challenge any of the purported teachings or assertions made in the last action at any appropriate time in the future, should the need arise. Furthermore, to the extent that the Examiner has relied on any Official Notice, explicitly or implicitly, Applicant specifically requests that the Examiner provide references supporting the teachings officially noticed, as well as the required motivation or suggestion to combine the relied upon notice with the other art of record.

In the event that the Examiner finds remaining impediment to a prompt allowance of this application that may be clarified through a telephone interview, the Examiner is requested to contact the undersigned attorney at (801) 533-9800.

Dated this 6th day of June, 2007.

Respectfully submitted,

RICK D. NYDEGGER
Registration No. 28,651
JENS C. JENKINS
Registration No. 44,803
COLBY C. NUTTALL
Registration No. 58,146
Attorneys for Applicant
Customer No. 047973

RDN:JCJ:CCN:gd
GD0000001889V001